1   **Agentive Representation in Mobile Services**

2

3   The present invention relates to the use of agents to

4   provide persistent, tailored presence in the electronic

5   world for a given user of a (suite of) mobile device(s),

6   in particular, a modular architecture of the agent and

7   messaging methods within and between agents.

8

9   A user has multiple presences in the electronic world,

10 including:

11 • the transient, anonymous presence of an online search;

12 • persistent occasional presence of online shopping at a

13   particular store;

14 • persistent passive presence of directed marking;

15 • persistent though temporary realtime presence in an

16   online game;

17

18 and many more. It would be advantageous to bring these

19 many applications and domains together, and provide the

20 user with a single, tailored interface to the electronic

21 world.

22

1 As users interact with the electronic world increasingly

2 frequently to serve an ever-greater set of goals, they

3 encounter three problems. First, the volume of

4 information can make it extremely difficult to identify

5 relevant sources: this is the well-known information

6 overload problem. Secondly, interacting with numerous

7 services (information provision, e-shopping, electronic

8 auction houses, alerting services, etc.) means that users

9 have to remember how to use a wide variety of different

10 interfaces, each with their own idiosyncrasies, required

11 data, stored data, and so on. Many web sites will

12 remember little or no information about given customers

13 other than their order history. This is the interface

14 problem. Finally, there is no structured way for these

15 services to interact. Booking a holiday for example,

16 would require visits to numerous web sites (information

17 provision, flight booking, hotel booking, newsgroup

18 archives, etc.) and often - indeed, usually - it is

19 simpler just to call a human travel agent. This is the

20 interaction problem. There are existing attempts to solve

21 each of these problems separately. These attempts have

22 had varying degrees of success and are at varying levels

23 of maturity: some web browsers, for example have built-in

24 components to try to tackle information overload though

25 for the most part these are not terribly effective;

26 similarly, web services offer a potential means of

27 integrating different services, but their deployment has

28 been limited to date, and it is not clear that there is

29 sufficient market pressure to further encourage providers

30 to provide web service based interaction.

31

32 Agentive representation offers a coherent means of

33 dealing with all three problems. Agents can act as

1 bidirectional filters of information, limiting

2 information presented to a user based on an internal user

3 model, and limiting information about the user that is

4 provided to electronic services based on internal rules

5 developed in conjunction with the user. This is a means

6 of tackling the information overload problem. Agents can

7 maintain information about dealing with other online

8 services, automating the process of form-filling, button-

9 clicking, and interaction with specific Web Services.

10 This offers a means of tackling the interface problem.

11 Finally, agents can act autonomously to collate

12 information and services in order to meet goals specified

13 by the user or adopted independently by the agent. This

14 offers a means of dealing with the interaction problem.

15

16 The idea of employing agents to represent users has been

17 widely deployed in systems in a variety of domains.

18 Typically, these systems are locked in to their

19 respective domains (such as e-commerce, stock trading

20 information, etc.), and do not try to cater for multiple

21 domains. They are also not fundamentally based on the

22 mobility of users (though some may have simple mobile

23 capabilities, such as SMS alerting). Indeed these two

24 restrictions - single domain and non-mobile - are

25 related. It would be advantageous to focus on the user,

26 wherever they may be, and whatever they may be doing,

27 rather than viewing a user as simply that part of a human

28 that is interacting with a particular computer system.

29

30 International Patent Application Number WO0157724

31 discloses having an agent represent a user that connects

32 via a mobile device. It fails at overcoming the above-

33 identified problems in two main respects. First, all

1 functionality is hardcoded, with no capacity for

2 concurrent and dynamic activity in multiple domains.

3 Second, the user connects to his or her agent via one

4 particular communication channel. It would be

5 advantageous for connection to be achieved through any

6 number of channels, mobile or wired, with media provided

7 by the agent for the user tailored to the device

8 currently in use.

9

10 It is an object of the present invention to provide

11 improved calling of methods within an agent.

12

13 It is a further object of the present invention to

14 provide improved messaging between agents and between

15 agents and users.

16 In accordance with a first aspect of the present

17 invention there is provided computing means having a

18 software agent for representing a person in the virtual

19 environment, the software agent comprising:

20 one or more application specific modules each of which

21 represents application specific features of the agent;

22 a core module which contains one or more functional

23 groups which define common or generic features of the

24 agent, said features at least in part facilitating inter-

25 agent communication, such that inter-agent communication

26 supports communication between a combination of the one

27 or more application specific module and the core module.

28

29 Preferably, the functionality of the functional group

30 comprises one or more of the following, belief

31 management, user profile management, agent-user

32 communication, module management, basic generic reasoning

1   tools and/or between agent module to module

2   communication.

3

4   Preferably, the core module is provided with method means

5   which provide the one or more functional groups.

6

7   Preferably, the functionality of the functional group

8   correspond to a set of labels.

9

10 Preferably, communication means are provided to

11 facilitate communication between application specific

12 modules in different agents.

13

14 Preferably, the core module acts as an interface between

15 external devices and the at least one application

16 specific module.

17

18 Preferably, specification of message conversation

19 protocols and the specification of primitive message

20 semantics are implemented in separate modules.

21

22 Preferably, the core module provides primitive semantics

23 for defining communication.

24

25 Preferably, the application specific module(s) specify

26 message conversation protocols.

27

28 Preferably, the software agent is further provided with

29 an inter-module communications means.

30

31 Preferably, said inter-module communications means

32 connects together all application specific modules and

33 the core module in the agent.

1

2   Preferably, the inter-module communication means is

3   provided with one or more function calls.

4

5   Preferably, the inter-module communication means provides

6   for communication between functions in different modules

7   of an agent.

8

9   Preferably, the inter-module communication means provides

10   for mapping a request from a first module to a method

11   means in a second module.

12

13   Preferably, said request from said first module comprises

14   a label specifying a function and said method means in a

15   second module corresponds to the specified function.

16

17   Preferably, the agent further comprises an address

18   resolving means for resolving an address in a message to

19   one of said plurality of modules.

20

21   Preferably said agent further comprises a transfer means

22   for transferring messages from said resolved modules such

23   that the messages are interleaved to allow an agent to be

24   simultaneously involved in multiple conversations with

25   other agents.

26

27   Preferably, the computing means is one or more computer.

28

29   Optionally, the computing means is one or more personal

30   digital assistant.

31

32   Optionally, the computing means is one or more mobile

33   communications device.

1

2   Optionally, the computing means is distributed across a

3   plurality of computing devices.

4

5   According to a second aspect of the invention there is

6   provided a method of performing functions in the software

7   agent in accordance with the first aspect of the

8   invention, the method comprising the steps of:

9   • receiving a request specifying a function;

10 • mapping said request to a module method corresponding

11   to the specified function; and

12 • invoking said module method.

13

14 Preferably said request comprises a label specifying said

15 function.

16

17 Preferably the step of invoking said module comprises the

18 steps of:

19 • receiving a request comprising a label;

20 • looking up the label in a table; and

21 • calling a method corresponding to the label.

22

23 Preferably the step of invoking said module further

24 comprises the step of selecting a highest priority method

25 corresponding to the label.

26

27 Optionally, the method of invoking said module further

28 comprises the step of returning a value to the originator

29 of the request.

30

31 According to a third aspect of the present invention,

32 there is provided a method of inter-agent communication

1 between agents as defined in the first aspect of the
2 invention, the method comprising the steps of:
3 • receiving a message comprising at least in part an
4   address from a first agent;
5 • resolving said address to one of a plurality of modules
6   in a second, receiving agent; and
7 • transferring the message to the resolved module.
8
9 Preferably said address specifies the module.
10
11 The method comprises the steps of communicating with an
12 external device by:
13 • identifying the device that a user is employing;
14 • mapping said device to a set of media types; and
15 • initiating the delivery of media to said device
16   responsive to the mapped set.
17
18 Optionally the method further includes the step of
19 limiting the set of media types based on user
20 preferences.
21
22 According to a fourth aspect of the present invention
23 there is provided a computer program comprising program
24 instructions for causing a computer to operate a software
25 agent as defined in the first aspect of the invention.
26
27 According to a fifth aspect of the present invention
28 there is provided a computer program comprising program
29 instructions for causing a computer to perform the method
30 as defined in the second aspect of the invention.
31
32 According to a fifth aspect of the present invention
33 there is provided a computer program comprising program

1   instructions for causing a computer to perform the method

2   as defined in the second aspect of the invention.

3

4   In order to provide a better understanding of the present

5   invention, an embodiment will now be described by way of

6   example only and with reference to the accompanying

7   Figures, in which:

8

9   Figure 1 illustrates, in schematic form, an agent in

10 accordance with a preferred embodiment of the present

11 invention;

12

13 Figure 2 illustrates, in schematic form, an overview of

14 agentive representation in a multi-service environment;

15

16 Figure 3 illustrates, in schematic form, the process by

17 which a label is resolved in accordance with a preferred

18 embodiment of the present invention;

19

20 Figure 4 illustrates, in schematic form, the process of a

21 module sending messages in accordance with the present

22 invention;

23

24 Figure 5 illustrates, in schematic form, the process of a

25 module receiving messages in accordance with the present

26 invention;

27

28 Figure 6 illustrates, in schematic form, conversation

29 interleaving in accordance with the present invention;

30

31 The inventions relate to an agent architecture and

32 methods for communication between modules in the agent,

1 with other agents in a multi-agent environment and with

2 users.

3

4 Although the embodiments of the invention described with

5 reference to the drawings comprise computer apparatus and

6 processes performed in computer apparatus, the invention

7 also extends to computer programs, particularly computer

8 programs on or in a carrier, adapted for putting the

9 invention into practice. The program may be in the form

10 of source code, object code, a code of intermediate

11 source and object code such as in partially compiled form

12 suitable for use in the implementation of the processes

13 according to the invention. The carrier may be any

14 entity or device capable of carrying the program.

15

16 For example, the carrier may comprise a storage medium,

17 such as ROM, for example a CD ROM or a semiconductor ROM,

18 or a magnetic recording medium, for example, floppy disc

19 or hard disc. .Further, the carrier may be a

20 transmissible carrier such as an electrical or optical

21 signal which may be conveyed via electrical or optical

22 cable or by radio or other means.

23

24 When the program is embodied in a signal which may be

25 conveyed directly by a cable or other device or means,

26 the carrier may be constituted by such cable or other

27 device or means.

28

29 Alternatively, the carrier may be an integrated circuit

30 in which the program is embedded, the integrated circuit

31 being adapted for performing, or for use in the

32 performance of, the relevant processes.

33

1   With reference to Figure 1, the architecture 100 of an

2   agent according to the present invention is best

3   visualised as including a torus. On the inside of the

4   torus 102, a special module, the core module 104,

5   attaches itself. On the outside of the torus, any number

6   of application specific modules 106, 108 may also become

7   attached. The security and unity of the agent is also

8   conceptually protected by a thin sphere 110 encompassing

9   all the modules. The torus itself coordinates all

10   communication between modules and between modules and

11   core: this is the Inter Module Communication Layer

12   (IMCL).

13

14   A user interacts with the electronic world for a host of

15   reasons in a wide variety of domains: entertainment, e-

16   commerce, professional, and so on. The present invention

17   provides a means of bringing together all of these tasks

18   and domains, and providing a single point of contact for

19   the user, and allowing the sharing of user data between

20   these different application domains. This contact is the

21   user's agent, both in the computer-science sense (where

22   agent oriented programming has particular restrictions,

23   techniques and approaches, and places particular demands

24   on software), and also in the intuitive sense of

25   providing services of advocacy and representation. A

26   user's agent is their permanent representative in the

27   electronic world. Ideally, each user has exactly one

28   agent, and a user's agent represents exactly one user (at

29   the very least, such a relationship exists in a given

30   context). The overall picture is as in Figure 2.

31

32   With reference to Figure 2, an overview of agentive

33   representation in a multiservice environment is shown.

12

1 The user 202 connects to their agent 206 at any time via

2 any device (2G phones, multimedia mobile handsets,

3 internet, etc.) in ways that are well known. The user

4 agents 204 which represent users in the virtual world are

5 shown. One user has a single agent 206 representing him

6 or her in all their interactions in the virtual world.

7 The service agents 208 provide specific services to any

8 agents that request them, or that the service agents

9 themselves decide to service. Information exchange

10 between user and service agents can be initiated from

11 either end. Some service agents 210 encapsulate existing

12 legacy services (e.g., databases, Web Services and

13 proprietary data handling systems). Broker agents 212

14 can mediate between a user and service agents. The user

15 agents service agents and broker agents may be provided

16 as a trusted service by a telecommunications operator.

17

18 An agent is a software entity with particular

19 characteristics. We refer here to software processes that

20 are:

21 (i)   persistent (in that they continue to exist for an

22   extended real time period, adapting to a single user

23   over that time);

24 (ii) proactive (in that they include not only reactive

25   behaviour, but also independently determined

26   behaviour);

27 (iii)   communicative (in that they communicate with

28   other agents); and

29 (iv) autonomous (in that they typically cannot be

30   directly modified by outside agencies, but must

31   instead be altered through communciation).

32

1   The user can communicate with his agent across

2   heterogeneous networks from a variety of devices,

3   including mobile handsets and internet clients. In

4   addition, however, the framework of the present invention

5   supports the transparent filtering of information

6   according to the device to which it is being sent. Thus

7   the components within an agent that initiate

8   communication with a user need not have any

9   representation of the device type a user is employing.

10 The content of the message is instead dynamically

11 tailored to the user's device (e.g. summary text to an

12 SMS-enabled mobile device, still pictures to a MMS-

13 enabled mobile device, streaming video to broadband

14 internet client platform, etc.).

15

16 The core is responsible for tailoring information to the

17 device that is known to currently be available to the

18 user. Thus, tailoring happens independently of the

19 module calls, so that individual modules do not need to

20 maintain device-specific information.

21

22 This filtering is achieved through a module-independent

23 communication object that is filled in by individual

24 modules when they need to communicate with the user.

25 This object has subparts for different forms of media

26 (text, picture, video, audio, etc). A module fills in as

27 many of these subparts as it is able. The core then

28 mediates the sending of that message to the user, by:

29 (i)   identifying which device the user is currently

30   employing (using a combination of historical usage

31   patterns, presence information, and most recent-

32   communication data);

1   (ii) mapping the device to a set of media types (so,

2   e.g., an old phone can handle text, a newer device,

3   pictures);

4   (iii)   further limiting the media types on the basis

5   of user preferences, and what has been made

6   available by the module; and

7   (iv) initiating the delivery of the appropriate media

8   from the user communication object constructed by

9   the module.

10

11 In order to provide representation for a user, an agent

12 must implement a range of functionality. This

13 functionality is gathered together into the core module.

14 Modules can safely make the assumption that the core is

15 available for them to make calls upon.

16

17 The core contains a range of specific methods that

18 implement particular components of functionality. These

19 methods can be grouped together into functional groups.

20 Thus the core can be subdivided into discrete areas of

21 functionality. Any module can make a call on any of the

22 methods in any of the areas of the core's functionality

23 via the IMCL. The core provides methods that provide

24 functionality corresponding to a fixed set of labels

25 concerned with generic agent activity. This functionality

26 includes:

27   1.Belief management (including lookup and update)

28   2.User profile management (including lookup and

29   update)

30   3.Agent-User communication

31   4.Module Management

32   5.Basic generic reasoning tools

1   6. Between-Agent Module-Module communication (BAMM)

2   (send and receive)

3

4   The agent as a whole is a unitary autonomous software

5   entity, and as such maintains a single, coherent set of

6   token expressions representing information about the

7   world. The language from which these beliefs are

8   constructed is given by domain-specific ontologies

9   provided centrally. Beliefs are stored in a single

10 database using existing technology.

11

12 The belief database is changed through the action of

13 methods in the core. These methods implement core labels

14 for belief update. Any module (including the core itself)

15 can make calls as described herein on these labels

16 through the IMCL.

17

18 Similarly, the belief database can be queried by any

19 method through a call to a label mapped through the IMCL

20 to core functionality. Thus a module can initiate update

21 or lookup on the currently held beliefs by calling this

22 label.

23

24 The user profile is a subset of the belief database, and

25 includes information specific to the user across a range

26 of domains. Again, the core implements labels

27 corresponding to update and query to the user profile.

28

29 There is the potential for the core to update the user

30 profile dynamically in response to user actions - that

31 is, the agent could adapt to and learn the user's

32 preferences as a result of repeated interaction.

33

1 User data (e.g., address; credit card details; age) and
2 user preferences (e.g., policy on releasing credit card
3 details; preference for aisle or window seat on planes;
4 preferred DVD supplier) are stored in a local, private,
5 secure database. Both user data and user preferences are
6 extracted in three ways. First, through an explicit
7 online interface that requests input on date of birth, or
8 supports update to reflect change of address. Second, if
9 the agent recognises information that it needs from the
10 user, it can ask for it directly (e.g. asking a yes/no
11 question by SMS). Third, as the user interacts with
12 services manually, the agent can intercept information
13 either explicitly or implicitly. If the user answers a
14 particular question from a particular online service, the
15 agent may either store that answer for future use, or ask
16 the user explicitly if such storage is appropriate or
17 useful. When acting autonomously, the agent provides only
18 the information that external service requires (and no
19 more), less anything that the user has placed a
20 restriction on. Thus, for example, when interacting with
21 an online newspaper, the newspaper provider may request
22 user registration, but not demand it. In this case, the
23 agent would provide no user information. Alternatively,
24 when interacting with a book e-tailer, the e-tailer may
25 require personal details including credit card data. If
26 the user has instructed his or her agent not to give out
27 credit card details without confirming it first, the
28 agent would halt interaction with that site until user
29 confirmation was sought and agreed.
30
31 These components could be represented by the steps:
32   1. Agent has goal of interacting with a service

1   2. Select required information from the user model

2   (UM) (accesses the UM)

3   3. Check that the user model permits all this

4   information to be freely given (accesses the UM)

5   If so,

6   4. Information given to the service

7   Otherwise

8   5. Process the restriction (either by terminating,

9   or by asking the user, or by performing some

10   other action)

11

12 The core also includes a subsystem responsible for

13 passing messages to, and receiving messages from the

14 user. The user may connect to his or her agent through a

15 number of different channels: using a web browser on a

16 PC, using a rich media mobile device (a Java phone, for

17 example), using a high capacity mobile device (such as

18 one that uses GPRS), or using an older, limited media

19 device (say that can only handle voice and SMS traffic).

20 The core implements labels that handle communication to

21 and from such devices quite transparently: the calling

22 module need not specify different communication types.

23

24 The means by which one agent communicates with another is

25 implemented in the core. Rather than supporting only

26 agent-to-agent messages, the architecture is instead

27 built around the idea that it is individual modules

28 within agents that communicate with one another (this is

29 "between agent module-module" or BAMM communication).

30 Thus a module with expertise in buying in a particular e-

31 commerce institution will communicate with a module in

32 another agent that has expertise in selling in that same

33 e-commerce institution. The fact that those agents also

1   happen to have modules with expertise in a range of other
2   diverse applications has no impact upon the conversation
3   between buyer and seller in this domain. It is thus
4   modules that structure conversations. The individual
5   utterances (or, more accurately, utterance types) that a
6   module uses to construct a given conversation are common
7   across the entire architecture. The sending and receiving
8   of these individual utterances is co-ordinated by the
9   core.
10
11   In this way, a module in an agent can conduct
12   conversations tailored to the domain in which the module
13   has competence. Though the conversation structure is
14   tailored, the implementation of primitive sending and
15   receiving is located in the core. This means that there
16   needs to be only one language definition - the language
17   that agents use for all communication. (If BAMM
18   communication was implemented solely in modules, those
19   modules would, by definition, use their own idiosyncratic
20   languages, and therefore the number of languages would be
21   proportional to the square of the number of module
22   types.) As language design and verification is a labour
23   intensive task, reducing the task by separating primitive
24   semantics from conversation definition, and rendering the
25   former once only in the core, saves a great deal of
26   effort.
27
28   The IMCL provides a small number of function calls, the
29   most important of which is the call which effects Within-
30   Agent Module-Module (WAMM) communication. When one module
31   wants to call a method in another module (including a
32   method provided by the core) it calls the IMCL's WAMM

1 communication method, passing it a label. The IMCL then

2 resolves that label by referring to its table of labels.

3

4 This means that one module need not know which other

5 module implements the functionality of a given label.

6 Indeed, a module can be implemented in such a way that it

7 can attempt a call on some labelled functionality, but

8 exhibits robustness in the event that no module is

9 present that implements that functionality. (Consider,

10 for example, module x that is, amongst other things,

11 responsible for performing some exponentiation

12 calculation. Module x has two ways of performing the

13 calculation – doing it itself, slowly and laboriously

14 using repeated addition, or by asking a specialised

15 module y that can do exponentiation quickly and

16 efficiently. The problem is that x has no way of knowing

17 whether or not y is installed. Thus x makes a call to the

18 IMCL requesting exponentiation on a particular data set.

19 If y is installed, the IMCL will pass the request to the

20 appropriate method within y. If y is not installed, the

21 IMCL will inform x that no module implements

22 exponentiation and x can then follow the more laborious

23 route of performing the calculation itself). The process

24 by which a label is resolved is summarised in Figure 3.

25

26 With reference to Figure 3, a module makes a call to

27 label L 310. The IMCL looks up L in a label table 312.

28 If L is not present 314, the IMCL returns "not found"

29 316. If L is present, and L does have multiple

30 resolutions 318, then the IMCL selects the highest

31 priority resolution 320. Next the IMCL calls the method

32 described in the resolution 322. Finally, when the

1   method returns a value 324, the IMCL passes the return
2   value back to the caller.
3
4   A practical advantage of the approach is that it removes
5   compile time dependencies: a module developer can design,
6   implement and test a module which makes calls to another
7   module that they do not have, or do not have access to,
8   or, indeed, that has not been developed at all. This
9   simplifies many of the problems of software engineering
10   in the large, and of multi-site collaborative development
11   work.
12
13   For sending messages, the core implements a unique label
14   that sends a preconstructed message that conforms to the
15   structure of the system's ACL through the transport layer
16   to the recipient agent. The series of steps by which this
17   is achieved is shown in Figure 4.
18
19   With reference to Figure 4, the components of the agent
20   102, 104, 106 and 110 are as described in Figure 1.
21   First the module builds an ACL message with module@agent
22   recipient and content 402. The module calls the IMCL
23   with a specific label (such as "talk2agent") and the ACL
24   message 404. IMCL resolves talk2agent label call to a
25   specific core method (such as "TalkToAgent")406. The
26   IMCL calls core's TalkToAgent method with the ACL message
27   408. core.TalkToAgent resolves agent name to transport
28   specific identifier 410. Transport calls are made to
29   deliver the message 412. Finally the message is
30   transported 414.
31
32   With reference to Figure 5, components of the agent 102,
33   104, 106 and 110 are as described in Figure 1. The

1   incoming message 502 corresponding to the outgoing
2   message 414 of Figure 4 is transported into the agent.
3   The message arrives in the core from the transport layer
4   504. The core makes a call 508 to the module's message
5   handler 510, from where the module processes the message.
6   For the receipt of ACL messages, the core implements a
7   queue mechanism. Individual messages should be addressed
8   to "module@agent", thus specifying not only the agent to
9   which the message is addressed, but also the specific
10   module within that agent. (Messages that are
11   underspecified and do not indicate a recipient module are
12   handled separately by the core). The core queues these
13   messages, and passes them to individual modules according
14   to the message address, when appropriate reprocessing
15   resources become available.
16
17   In line with a number of other frameworks, the semantics
18   of ACL utterances are defined in terms of preconditions
19   and postconditions - that is, things that must be true
20   before a message can be sent, and things that must be
21   true after a message has been received (for example,
22   inform-ing an agent may require that the fact being
23   informed is initially believed by the informing agent -
24   this is sincerity).
25
26   The core is responsible for implementing the ACL
27   semantics. The message sending functionality filters
28   messages, only sending those that meet the semantic
29   constraints (such as sincerity). The message receiving
30   functionality similarly implements the postcondition
31   semantics by updating the belief database before the
32   message is placed on the queue for handling by the
33   recipient module.

1

2   The combination of queuing mechanisms for messages,

3   explicit module addressing, and a common, core-

4   implemented semantics for primitives, provides for a

5   technique that may be called 'conversation interleaving'.

6

7   Conversation interleaving refers to the way in which a

8   single agent can simultaneously be involved in multiple

9   conversations with other agents, with individual modules

10 responsible for the maintenance of a given conversation,

11 even though the primitives from which conversations are

12 composed are sent and received through the agent's single

13 interface with the rest of the agent world.

14

15 By analogy, imagine yourself on the phone trying, say, to

16 arrange car insurance - every so often, the person you

17 are speaking to comes back to you, has a brief exchange

18 and then puts you back on hold while they try and find

19 another quote. Simultaneously you could be having a chat

20 with an office colleague. The 'car insurance' part of you

21 is holding a conversation on the phone, and the 'office

22 smalltalk' part with someone in front of you - two

23 simultaneous conversations even though you can only say

24 one thing to one person at a time. An example of

25 conversation interleaving is illustrated in Figure 6.

26

27 With reference to Figure 6, the agent 100 contains the

28 same components 102, 104, 106 and 108 as described in

29 Figure 1. The first module 106 send messages 602

30 destined for agent A 604 to the core 104. The second

31 module 108 send messages 606 destined for agent B 608 to

32 the core. The core functionality 610 marshals outgoing

33 messages and the messages are sent 612 to the transport

1   layer for delivery (as in Figure 4). Therefore the

2   messages 602 and 606 are interleaved 614 and messages

3   from the first module are delivered to agent A and

4   messages from the second module are delivered to agent B.

5

6